Title: AUTOMATIC PRICING METHOD AND DEVICE
Inventor(s): Naoki ABE, et al.
DOCKET NO.: 040373/0303

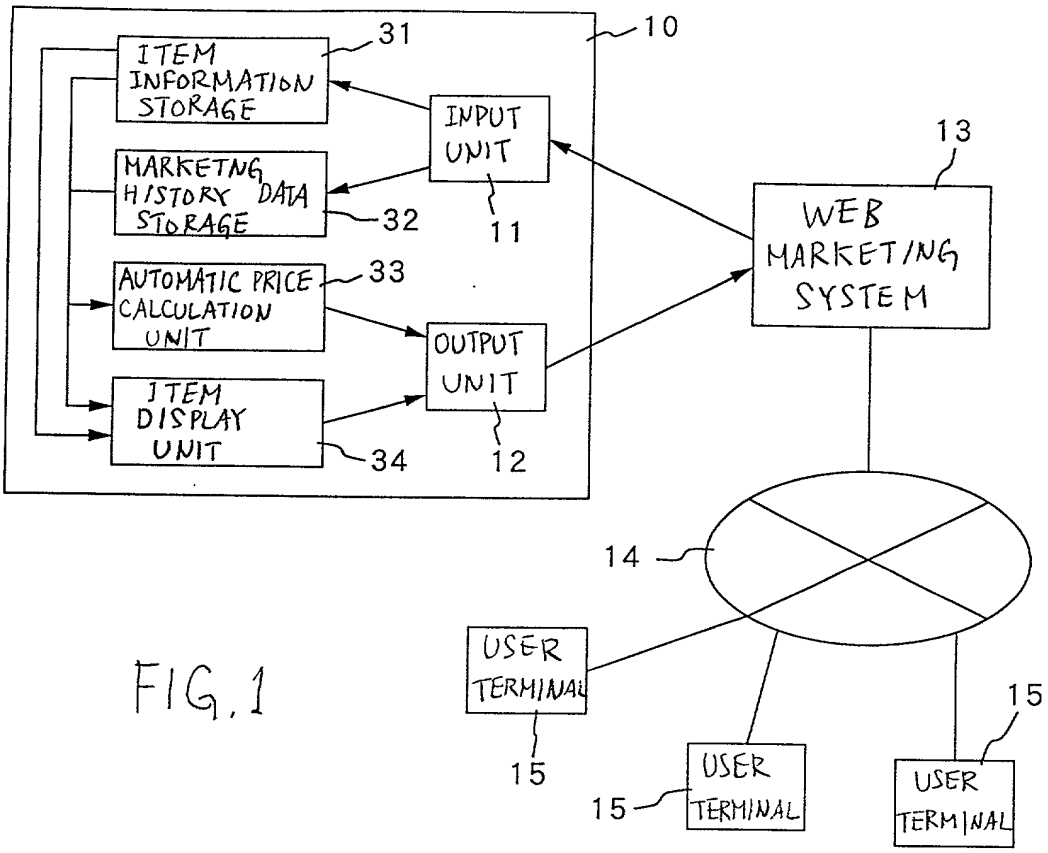

FIG. 1

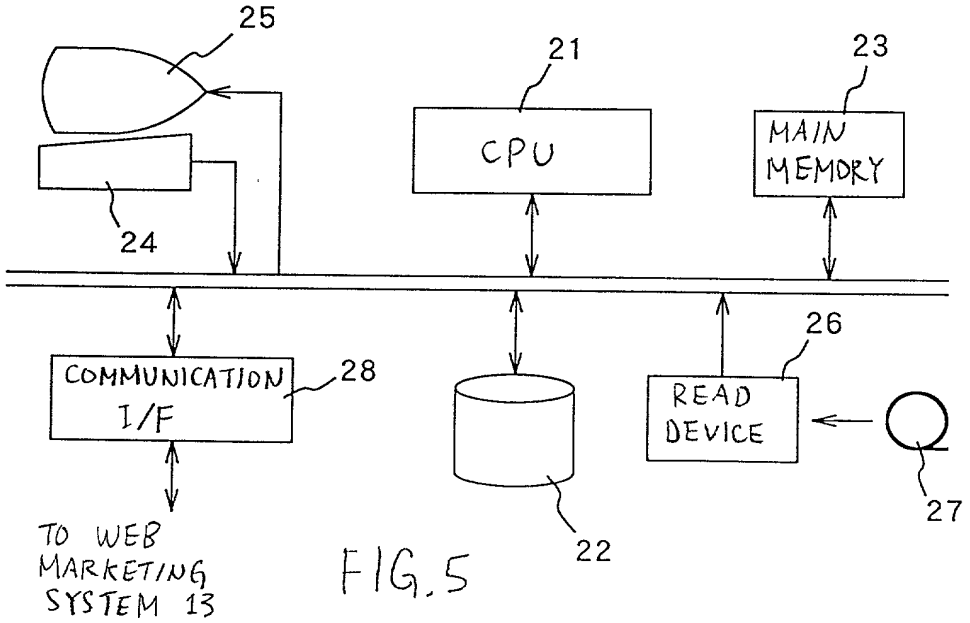

FIG. 5

Algorithm: ***StochPrice***
(P_{init}: Initial price, T: Unit sales period)

1. Initialization
   1.1. Initial price: $p := p_{init}$
   1.2. Initial time: $t :=$ current time
   1.3. Initial setting of trial number: $I := 1$
2. **Repeat For** $I = 1$ until forever
   2.1. Set Δ as follows: $\Delta := I^{-1/3}$
   2.2. For a period of T, set price to $p + \Delta$
   2.3. Let $S(p+\Delta)$ be the amount of sales during this time.
   2.4. For a period of T, set price to $p - \Delta$
   2.5. Let $S(p-\Delta)$ be the amount of sales during this time.
   2.6. Calculate the obtained profit as follows:
   $P(p + \Delta) = S(p + \Delta) \cdot (p + \Delta)$
   $P(p - \Delta) = S(p - \Delta) \cdot (p - \Delta)$
   2.7. Set the update interval A as follows:
   $$A := \frac{1}{I}$$
   2.8. Update the current price as follows:
   $$p := p + \frac{A}{\Delta} \frac{P(p + \Delta) - P(p - \Delta)}{2T}$$
   2.9. If necessary, clamp the value of p between the maximum and minimum possible prices.
   $p := \min\{p_{max}-\Delta, \max\{p_{min}+\Delta, p\}\}$
   2.10. Update current time t.
   $t := t + 2T$
   2.11. Store current time t, price p and sales $S(p)=S(p-\Delta)+S(p-\Delta)$ to the database.

FIG. 2

Algorithm: ***FeaturePrice***
(W_{init} : Initial weight vector, T : unit sales period)

1. Initialization
   1.1. Initial weight vector: $W := W_{init}$
   1.2. Initial time: t := current time
2. **Repeat For** I = 1 until forever
   2.1. **For** item i = 1 until N (number of item)
      2.1.1. $X(i)$:= attribute vector for item i
      2.1.2. $P(i) := W \cdot X(i)$
      2.1.3. $\Delta := I^{-1/3}$
      2.1.4. $\vec{V}(i)$ = Random-vector()
      2.1.5. $\vec{\Delta}(i) = \Delta \cdot (\vec{V}(i) / |\vec{V}(i)|)$
      2.1.6. Set current price for item i as follows:
      $p(i) := \{W + \vec{\Delta}(i)\} \cdot X(i)$
      2.1.7. For each item i, **If** $p(i)_{min} > (W + \vec{\Delta}(i) \cdot X(i))$ or $p(i)_{max} < (W + \vec{\Delta}(i) \cdot X(i))$ **then** select maximum positive constant Π which satisfies the following equation and put $\vec{\Delta}(i) := \Pi\vec{\Delta}(i)$:
      $p(i)_{min} \le (W + \Pi\vec{\Delta}(i)) \cdot X(i) \le p(i)_{max}$
   2.2. For a time period of T, set the price of each item (i) to $p(i) := (W + \vec{\Delta}(i)) \cdot X(i)$ and conduct sales.
   2.3. Let $S(W + \vec{\Delta}(i))$ be amount of sales thus obtained for each item (i).
   2.4. For a time period of T, set the price of each item (i) to $p(i) := (W - \vec{\Delta}(i)) \cdot X(i)$ and conduct sales.
   2.3. Let $S(W - \vec{\Delta}(i))$ be amount of sales thus obtained for each item (i).
   2.6. For each item i, calculate total profits based on the above amount of sales.
   $P(W + \vec{\Delta}(i)) = S(W + \vec{\Delta}(i)) \cdot X(i)\ (W + \vec{\Delta}(i))$
   $P(W - \vec{\Delta}(i)) = S(W - \vec{\Delta}(i)) \cdot X(i)\ (W - \vec{\Delta}(i))$
   2.7. **For** i = 1 until number of items
   Update the weight vector W as follows:
   $$W := W + \frac{A}{|\vec{\Delta}(i)|} \frac{P(W + \vec{\Delta}(i)) - P(W - \vec{\Delta}(i))}{2T}$$
   2.8. Update current time t.
   $t := t + 2T$
   2.9. Store current time t, price p and $S(p) = S(W - \vec{\Delta}(i)) + S(W + \vec{\Delta}(i))$ to the database.

FIG. 3

Algorithm: ***VarietySelection***
(W : weight vector, G : set of items, n : number of items to be displayed, N : Number of iterations)

1. Initialization
   1.1. Sort G in increasing order of $PTotal(i,W)$
   1.2. S := First - $n(G, n)$
   1.3. $\overline{S} := G \backslash S$
2. **Repeat for** i = 1 until N
   2.1. Randomly select item $j \in S$.
   2.2. If there exists item k such that exchanging j, k would result in increasing the evaluation value of $\sum_{i \in S} \lambda_1 PTotal(i,W) + \lambda_2 H(S)$
   then make that exchange and update S and $\overline{S}$.
3. Output S.

FIG. 4